CS4670/5760: Computer Vision

Kavita Bala

Lecture 17: RANSAC
Alignment

• Alignment: find parameters of model that maps one set of points to another

• Typically want to solve for a global transformation that accounts for *most* true correspondences

• Difficulties
  – Noise (typically 1-3 pixels)
  – Outliers (often 50%)
Least squares: find $t$ to minimize

$$||At - b||^2$$

$$t^T (A^T A)t - 2t^T (A^T b) + ||b||^2$$

- To solve, form the *normal equations*
  - Differentiate and equate to 0 to minimize

$$A^T A t = A^T b$$

$$t = \left( A^T A \right)^{-1} A^T b$$
Affine transformations

- Matrix form

\[
\begin{bmatrix}
    x_1 & y_1 & 1 & 0 & 0 & 0 \\
    0 & 0 & 0 & x_1 & y_1 & 1 \\
    x_2 & y_2 & 1 & 0 & 0 & 0 \\
    0 & 0 & 0 & x_2 & y_2 & 1 \\
    \vdots \\
    x_n & y_n & 1 & 0 & 0 & 0 \\
    0 & 0 & 0 & x_n & y_n & 1 \\
\end{bmatrix}
\begin{bmatrix}
    a \\
    b \\
    c \\
    d \\
    e \\
    f \\
\end{bmatrix}
= 
\begin{bmatrix}
    x'_1 \\
    y'_1 \\
    x'_2 \\
    y'_2 \\
    \vdots \\
    x'_n \\
    y'_n \\
\end{bmatrix}
\]

\[
A_{2n \times 6} t_{6 \times 1} = b_{2n \times 1}
\]
Solving for homographies

\[
\begin{bmatrix}
  x'_i \\
  y'_i \\
  1
\end{bmatrix} = \begin{bmatrix}
  h_{00} & h_{01} & h_{02} \\
  h_{10} & h_{11} & h_{12} \\
  h_{20} & h_{21} & h_{22}
\end{bmatrix} \begin{bmatrix}
  x_i \\
  y_i \\
  1
\end{bmatrix}
\]

\[
x'_i = \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}}
\]

\[
y'_i = \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}}
\]

\[
x'_i(h_{20}x_i + h_{21}y_i + h_{22}) = h_{00}x_i + h_{01}y_i + h_{02}
\]

\[
y'_i(h_{20}x_i + h_{21}y_i + h_{22}) = h_{10}x_i + h_{11}y_i + h_{12}
\]
Solving for homographies

\[
x'_i(h_{20}x_i + h_{21}y_i + h_{22}) = h_{00}x_i + h_{01}y_i + h_{02}
\]
\[
y'_i(h_{20}x_i + h_{21}y_i + h_{22}) = h_{10}x_i + h_{11}y_i + h_{12}
\]

\[
\begin{bmatrix}
x_i & y_i & 1 & 0 & 0 & 0 & -x'_i x_i & -x'_i y_i & -x'_i \\
0 & 0 & 0 & x_i & y_i & 1 & -y'_i x_i & -y'_i y_i & -y'_i \\
\end{bmatrix}
\begin{bmatrix}
h_{00} \\
h_{01} \\
h_{02} \\
h_{10} \\
h_{11} \\
h_{12} \\
h_{20} \\
h_{21} \\
h_{22} \\
\end{bmatrix} = \begin{bmatrix}
0 \\
0 \\
\end{bmatrix}
\]
Solving for homographies

Defines a least squares problem: \[
\minimize \| Ah - 0 \|^2
\]

- Since \( h \) is only defined up to scale, solve for unit vector \( \hat{h} \)
- Solution: \( \hat{h} = \) eigenvector of \( A^T A \) with smallest eigenvalue
- Works with 4 or more points
# Recap: Two Common Optimization Problems

<table>
<thead>
<tr>
<th>Problem statement</th>
<th>Solution</th>
</tr>
</thead>
</table>
| \[
\text{minimize } \| \mathbf{Ax} - \mathbf{b} \|^2
\]
least squares solution to \( \mathbf{Ax} = \mathbf{b} \) | \[
\mathbf{x} = \left( \mathbf{A}^T \mathbf{A} \right)^{-1} \mathbf{A}^T \mathbf{b}
\]
\( \mathbf{x} = \mathbf{A} \backslash \mathbf{b} \) (matlab) |

<table>
<thead>
<tr>
<th>Problem statement</th>
<th>Solution</th>
</tr>
</thead>
</table>
| \[
\text{minimize } \mathbf{x}^T \mathbf{A}^T \mathbf{A} \mathbf{x} \quad \text{s.t.} \quad \mathbf{x}^T \mathbf{x} = 1
\]
non-trivial lsq solution to \( \mathbf{Ax} = 0 \) | \[
[\mathbf{v}, \lambda] = \text{eig}(\mathbf{A}^T \mathbf{A})
\]
\( \lambda_1 < \lambda_{2..n} \): \( \mathbf{x} = \mathbf{v}_1 \) |
Fig. 2.4. **Removing perspective distortion.** (a) The original image with perspective distortion – the lines of the windows clearly converge at a finite point. (b) Synthesized frontal orthogonal view of the front wall. The image (a) of the wall is related via a projective transformation to the true geometry of the wall. The inverse transformation is computed by mapping the four imaged window corners to corners of an appropriately sized rectangle. The four point correspondences determine the transformation. The transformation is then applied to the whole image. Note that sections of the image of the ground are subject to a further projective distortion. This can also be removed by a projective transformation.
Hybrid Image competition results

• **Hall of Fame**: http://www.cs.cornell.edu/courses/cs4670/2016sp/artifacts/pa1/hof.html
Runners Up
Benjamin Siper, Sania Nagpal
Nadav Nehoran, Thomas Ilyevsky
Daniel Donenfeld, Markus Salasoo
Joshua Chan, Jerica Huang
Third Place
Aditi Jain, Ross Tannenbaum
James Briggs, Vishwanathan Ramanathan
Second Place
Aaron Ferber, Mateo Espinosa Zarlenga
First Place
Mingyang Li, Yuan Huang
Fitting and Alignment

Fitting: find the parameters of a model that best fit the data

Alignment: find the parameters of the transformation that best align matched points
Least squares: linear regression

\[ y = mx + b \]
Linear regression

\[ \text{Cost}(m, b) = \sum_{i=1}^{n} |y_i - (mx_i + b)|^2 \]
Linear regression

\[
\begin{bmatrix}
  x_1 & 1 \\
  x_2 & 1 \\
  \vdots & \vdots \\
  x_n & 1 \\
\end{bmatrix}
\begin{bmatrix}
  m \\
  b \\
\end{bmatrix}
= 
\begin{bmatrix}
  y_1 \\
  y_2 \\
  \vdots \\
  y_n \\
\end{bmatrix}
\]
Image Alignment Algorithm

Given images A and B

1. Compute image features for A and B
2. Match features between A and B
3. Compute homography between A and B using least squares on set of matches

What could go wrong?
Outliers
Robustness

Problem: Fit a line to these datapoints

Least squares fit
Idea

• Given a hypothesized line
• Count the number of points that “agree” with the line
  – “Agree” = within a small distance of the line
  – I.e., the **inliers** to that line

• For all possible lines, select the one with the largest number of inliers
Counting inliers
Counting inliers

Inliers: 3
Counting inliers

Inliers: 20
How do we find the best line?

• Unlike least-squares, no simple closed-form solution

• Hypothesize-and-test
  – Try out many lines, keep the best one
  – Which lines?
RANSAC (Random Sample Consensus)

Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

**Repeat** 1-3 until the best model is found with high confidence
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

**Repeat** 1-3 until the best model is found with high confidence
RANSAC

Line fitting example

\[ N_I = 6 \]

Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

**Repeat** 1-3 until the best model is found with high confidence
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence
RANSAC

• Idea:
  – All the inliers will agree with each other on the translation vector; the (hopefully small) number of outliers will (hopefully) disagree with each other
    • RANSAC only has guarantees if there are < 50% outliers

  – “All good matches are alike; every bad match is bad in its own way.”
    – Tolstoy via Alyosha Efros
Translations
RAndom SAmple Consensus

Select *one* match at random, count *inliers*
RANDOM SAMPLE Consensus

Select another match at random, count inliers
**Random Sample Consensus**

Output the translation with the highest number of inliers
Final step: least squares fit

Find average translation vector over all inliers
RANSAC

• **Inlier threshold** related to the amount of noise we expect in inliers
  – Often model noise as Gaussian with some standard deviation (e.g., 3 pixels)

• **Number of rounds** related to the percentage of outliers we expect, and the probability of success we’d like to guarantee
  – Suppose there are 20% outliers, and we want to find the correct answer with 99% probability
  – How many rounds do we need?
How many rounds?

• If we have to choose \( k \) samples each time
  – with an inlier ratio \( p \)
  – and we want the right answer with probability \( P \)

<table>
<thead>
<tr>
<th>( k )</th>
<th>95%</th>
<th>90%</th>
<th>80%</th>
<th>75%</th>
<th>70%</th>
<th>60%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>11</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>19</td>
<td>35</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>13</td>
<td>17</td>
<td>34</td>
<td>72</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>6</td>
<td>12</td>
<td>17</td>
<td>26</td>
<td>57</td>
<td>146</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>7</td>
<td>16</td>
<td>24</td>
<td>37</td>
<td>97</td>
<td>293</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>8</td>
<td>20</td>
<td>33</td>
<td>54</td>
<td>163</td>
<td>588</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>9</td>
<td>26</td>
<td>44</td>
<td>78</td>
<td>272</td>
<td>1177</td>
</tr>
</tbody>
</table>

\( P = 0.99 \)

Source: M. Pollefeys
To ensure that the random sampling has a good chance of finding a true set of inliers, a sufficient number of trials $S$ must be tried. Let $p$ be the probability that any given correspondence is valid and $P$ be the total probability of success after $S$ trials. The likelihood in one trial that all $k$ random samples are inliers is $p^k$. Therefore, the likelihood that $S$ such trials will all fail is

$$1 - P = (1 - p^k)^S$$  \hspace{1cm} (6.29)

and the required minimum number of trials is

$$S = \frac{\log(1 - P)}{\log(1 - p^k)}.$$  \hspace{1cm} (6.30)

<table>
<thead>
<tr>
<th>proportion of inliers $p$</th>
<th>k</th>
<th>95%</th>
<th>90%</th>
<th>80%</th>
<th>75%</th>
<th>70%</th>
<th>60%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>11</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>19</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>13</td>
<td>17</td>
<td>34</td>
<td>72</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4</td>
<td>6</td>
<td>12</td>
<td>17</td>
<td>26</td>
<td>57</td>
<td>146</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>4</td>
<td>7</td>
<td>16</td>
<td>24</td>
<td>37</td>
<td>97</td>
<td>293</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>4</td>
<td>8</td>
<td>20</td>
<td>33</td>
<td>54</td>
<td>163</td>
<td>588</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>5</td>
<td>9</td>
<td>26</td>
<td>44</td>
<td>78</td>
<td>272</td>
<td>1177</td>
</tr>
</tbody>
</table>

$P = 0.99$
How big is $k$?

- For alignment, depends on the motion model
  - Here, each sample is a correspondence (pair of matching points)

<table>
<thead>
<tr>
<th>Name</th>
<th>Matrix</th>
<th># D.O.F.</th>
<th>Preserves:</th>
<th>Icon</th>
</tr>
</thead>
<tbody>
<tr>
<td>translation</td>
<td>$[ I \mid t ]_{2\times3}$</td>
<td>2</td>
<td>orientation + ⋯</td>
<td></td>
</tr>
<tr>
<td>rigid (Euclidean)</td>
<td>$[ R \mid t ]_{2\times3}$</td>
<td>3</td>
<td>lengths + ⋯</td>
<td></td>
</tr>
<tr>
<td>similarity</td>
<td>$[ sR \mid t ]_{2\times3}$</td>
<td>4</td>
<td>angles + ⋯</td>
<td></td>
</tr>
<tr>
<td>affine</td>
<td>$[ A ]_{2\times3}$</td>
<td>6</td>
<td>parallelism + ⋯</td>
<td></td>
</tr>
<tr>
<td>projective</td>
<td>$[ \tilde{H} ]_{3\times3}$</td>
<td>8</td>
<td>straight lines</td>
<td></td>
</tr>
</tbody>
</table>
RANSAC pros and cons

• Pros
  – Simple and general
  – Applicable to many different problems
  – Often works well in practice

• Cons
  – Parameters to tune
  – Sometimes too many iterations are required
  – Can fail for extremely low inlier ratios
RANSAC

• An example of a “voting”-based fitting scheme
• Each hypothesis gets voted on by each data point, best hypothesis wins

• There are many other types of voting schemes
  – E.g., Hough transforms...
Hough transform


Given a set of points, find the curve or line that explains the data points best

\[ y = m x + b \]
Hough Transform: Outline

1. Create a grid of parameter values

2. Each point votes for a set of parameters, incrementing those values in grid

3. Find maximum or local maxima in grid
Hough transform

Slide from S. Savarese
\[ d = x \cos \theta + y \sin \theta \]
Hough transform
Fitting Summary

• Least Squares Fit
  – closed form solution
  – robust to noise
  – not robust to outliers

• Hough transform
  – robust to noise and outliers
  – can fit multiple models
  – only works for a few parameters (1-4 typically)

• RANSAC
  – robust to noise and outliers
  – works with a moderate number of parameters (e.g., 1-8)